Team SWEEPer: Joint Sentence Extraction and Fact Checking with Pointer Networks

Christopher Hidey[*]
Department of Computer Science
Columbia University
New York, NY 10027
chidey@cs.columbia.edu

Mona Diab
Amazon AI Lab
diabmona@amazon.com

Abstract

Many tasks such as question answering and reading comprehension rely on information extracted from unreliable sources. These systems would thus benefit from knowing whether a statement from an unreliable source is correct. We present experiments on the FEVER (Fact Extraction and VERification) task, a shared task that involves selecting sentences from Wikipedia and predicting whether a claim is supported by those sentences, refuted, or there is not enough information. Fact checking is a task that benefits from not only asserting or disputing the veracity of a claim but also finding evidence for that position. As these tasks are dependent on each other, an ideal model would consider the veracity of the claim when finding evidence and also find only the evidence that is relevant. We thus jointly model sentence extraction and verification on the FEVER shared task. Among all participants, we ranked 5th on the blind test set (prior to any additional human evaluation of the evidence).

1 Introduction

Verifying claims using textual sources is a difficult problem, requiring natural language inference as well as information retrieval if the sources are not provided. The FEVER task (Thorne et al., 2018) provides a large annotated resource for extraction of sentences from Wikipedia and verification of the extracted evidence against a claim. A system that extracts and verifies statements in this framework must consist of three components: 1) Retrieving Wikipedia articles. 2) Identifying the sentences from Wikipedia that support or refute the claim. 3) Predicting supporting, refuting, or not enough info.

We combine these components into two stages: 1) identifying relevant documents (Wikipedia articles) for a claim and 2) jointly extracting sentences from the top-ranked articles and predicting a relation for whether the claim is supported, refuted, or if there is not enough information in Wikipedia. We first identify relevant documents by ranking Wikipedia articles according to a model using lexical and syntactic features. Then, we derive contextual sentence representations for the claim paired with each evidence sentence in the extracted documents. We use the ESIM module (Chen et al., 2017b) to create embeddings for each claim/evidence pair and use a pointer network (Vinyals et al., 2015) to recurrently extract only relevant evidence sentences while predicting the relation using the entire set of evidence sentences. Finally, given these components, which are pre-trained using multi-task learning (Le et al., 2016), we tune the parameters of the entailment component given the extracted sentences.

Our experiments reveal that jointly training the model provides a significant boost in performance, suggesting that the contextual representations learn information about which sentences are most important for relation prediction as well as information about the type of relationship the evidence has to the claim.

2 Document Retrieval

For the baseline system, the document retrieval component from DrQA (Chen et al., 2017a) for $k = 5$ documents only finds the supporting evidence 55% of the time. This drops to 44% using the same model for sentence retrieval at $l = 5$ sentences. In comparison, in the original work of Chen et al. (2017a), they find a recall of 70-86% for all tasks with $k = 5$. This is partly due to the misleading information present in the false claims, whereas for question answering, the question is not designed adversarially to contain con-

[*] Work completed while at Amazon AI Lab

Proceedings of the First Workshop on Fact Extraction and VERification (FEVER), pages 150–155
Brussels, Belgium, November 1, 2018. ©2018 Association for Computational Linguistics

tradicting information. Examining the supporting and refuting claims in isolation, we find that document retrieval at $k = 5$ is 59% and 49% respectively and sentence selection at $l = 5$ is 49% and 39%. For example, the false claim ”Murda Beatz was born on February 21, 1994.” (his birth date is February 11) also retrieves documents for people born on February 21. In the question answering scenario, an example question might be ”Which rap artist was born on February 11, 1994 in Fort Erie, Ontario?” which allows an IR system to return documents using the disambiguating n-grams about his location and place of birth.

This motivates the decision to focus on noun phrases. Many of the claims contain the correct topic of the Wikipedia article in the subject position of both the claim and either the first or second sentence. When the topic is not in the first sentence, it is often because the title is ambiguous and the first sentence is a redirect. For example, for the article “Savages (2012 film)” the first two sentences are “For the 2007 film, see The Savages. Savages is a 2012 American crime thriller film directed by Oliver Stone.”

We thus parse Wikipedia and all the claims using CoreNLP (Manning et al., 2014) and train a classifier with the following lexical and syntactic features from the claim and the first two sentences in the Wikipedia article:

- TF/IDF from DrQA for full article

- Overlap between the subject/object/modifier in the claim with the subject/object/modifier in the first and second sentence of Wikipedia. The topic of the article is often the subject of the sentence in Wikipedia, but it is occasionally a disambiguating modifier such as “also known as”. The topic of the claim is often in the subject position as well. We also add overlap between named entities (in case the parsing fails) and upper case words (in case the parsing and NER fails). For each of subject/object/modifier/upper/entity, for both upper and lower case, we consider the coverage of the claim from the first two Wikipedia sentences, for $25 * 2 * 2 = 100$ features.

- Average/max/minimum of GloVe (Pennington et al., 2014) embeddings in the claim and the first and second sentence of Wikipedia. This feature captures the type of sentence. If it is a person, it may have words like ’born’, ’known as’, etc. and if it is a disambiguation sentence it should have words like ’refers’ or ’see’ or ’confused’. It also allows for the handling of cases where the sentence tokenization splits the first sentence.

- Features based on the title: whether the title is completely contained in the claim and the overlap between the claim and the title (with and without metadata information in parentheses), for both upper and lower case.

Model	Recall at $k = 5$	
	Dev	Test
Baseline (DrQA)	55.3	54.2
MLP without title features	82.7	79.7
MLP with title features	90.7	90.5

Figure 1: Document Retrieval Results

2.1 Experiments

We train a Multi-Layer Perceptron (MLP) using Pytorch (Paszke et al., 2017) on the top 1000 documents extracted using DrQA, which obtains a recall of 95.3%, with early stopping on the paper development set. We used the Adam optimizer (Kingma and Ba, 2015) with a learning rate of 0.001 and hyper-parameters of $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\epsilon = 10^{-8}$. We used pre-trained 300-dimensional GloVe embeddings. We clip gradients at 2 and use a batch size of 32.

2.2 Results

Document recall at $k = 5$ articles is presented in Figure 1. We present results on the paper development and test sets as the evidence for the shared task blind test set was unavailable as of this writing. The model with lexical and syntactic features obtains around 25 points absolute improvement over the DrQA baseline and the title features when added provide an additional 8 points improvement on the development set and 10.8 points on test.

3 Joint Sentence Extraction and Relation Prediction

Recurrent neural networks have been shown to be effective for extractive summarization (Nallapati et al., 2017; Chen and Bansal, 2018). However, in this context we want to extract sentences that also help predict whether the claim is supported or refuted so we jointly model extraction and relation prediction and train in a multi-task setting.

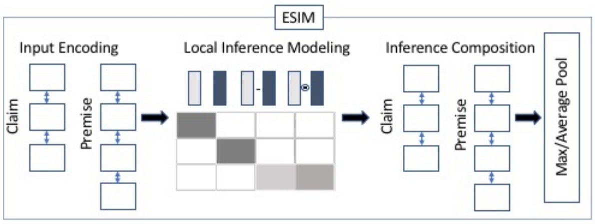

Figure 2: The contextual claim and evidence sentence representations obtained with ESIM.

First, given the top k extracted documents from our retrieval component, we select the top l sentences using a weighted combination of Jaccard similarity and cosine similarity of average word embeddings. Hyper-parameters were tuned so that the model would have minimal difference in recall from the document retrieval stage but still fit in memory on a Tesla V100 with 16GB of memory. We selected $k = 5$ and $l = 50$, with Jaccard similarity weighted at 0.3 and GloVe embeddings weighted at 0.7. We found that recall on the development set was 90.3.

We then store contextual representations of the claim and each evidence sentence in a memory network and use a pointer network to extract the top 5 sentences sequentially. Simultaneously, we use the entire memory of up to l sentences to predict the relation: supports, refutes, or not enough info.

3.1 Sentence Representation

For each sentence in the evidence, we create contextual representations using the ESIM module (Chen et al., 2017b), which first encodes the claim and the evidence sentence using an LSTM, attends to the claim and evidence, composes them using an additional LSTM layer, and finally applies max and average pooling over time (sentence length) to obtain a paired sentence representation for the claim and evidence. This representation is depicted in Figure 2. For more details, please refer to (Chen et al., 2017b).

The representation for a claim c and extracted evidence sentence e_p for c is then:

$$m_p = ESIM(c, e_p) \tag{1}$$

3.2 Sentence Extraction

Next, to select the top 5 evidence sentences, we use a pointer network (Vinyals et al., 2015; Chen and Bansal, 2018) over the evidence for claim c to extract sentences recurrently. The extraction probability[1] for sentence e_p at time $t < 5$ is then:

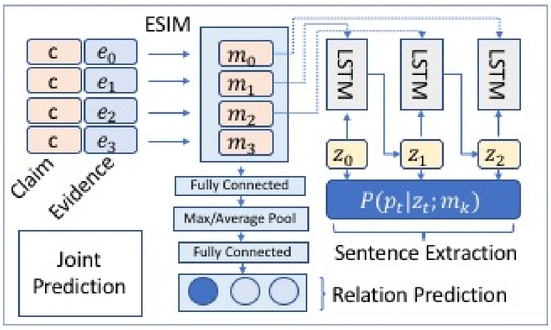

Figure 3: Our multi-task learning architecture with contextual ESIM representations used for evidence sentence extraction via a pointer network and relation prediction with max/average pooling and dense layers.

$$u_p^t = \begin{cases} v_e^T \tanh(W[m_p; h^{t,q}]), & \text{if} p_t \neq p_s \forall s < t. \\ -\inf, & \text{otherwise.} \end{cases} \tag{2}$$

$$P(p_t | p_0 \cdots p_{t-1}) = \text{softmax}(u^t) \tag{3}$$

with $h^{t,q}$ computed using the output of q hops over the evidence (Vinyals et al., 2016; Sukhbaatar et al., 2015):

$$\alpha^{t,o} = \text{softmax}(v_h^T \tanh(W_{g1} m_p + W_{g2} h^{t,o-1})) \tag{4}$$

$$h^{t,o} = \sum_j \alpha^{t,o} W_{g1} m_j \tag{5}$$

At each timestep t, we update the hidden state z_t of the pointer network LSTM. Initially, $h^{t,0}$ is set to z_t. We train and validate the pointer network using the extracted top l sentences. For all training examples, we randomly replace evidence sentences with gold evidence if no gold evidence is found.

3.3 Relation Prediction

In order to predict the support, refute, or not enough info relation, we use a single-layer MLP to obtain an abstract representation of the sentence representation used for extraction, then apply average and max pooling over the contextual representations of the claim and evidence sentences to obtain a single representation m for the entire memory. Finally, we use a 2-layer MLP to predict this relation given m. The entire joint architecture is presented in Figure 3.

[1] Set to $-inf$ only while testing

3.4 Optimization

We train the model to minimize the negative log likelihood of the extracted evidence sequence[2] and the cross-entropy loss ($\mathcal{L}(\theta_{rel})$) of the relation prediction. The pointer network is trained as in a sequence-to-sequence model:

$$\mathcal{L}(\theta_{ptr}) = -1/T \sum_{t=0..T} \log P_{\theta_{ptr}}(p_t|p_{0:t-1}) \tag{6}$$

and the overall loss is then:

$$\mathcal{L}(\theta) = \lambda\mathcal{L}(\theta_{ptr}) + \mathcal{L}(\theta_{rel}) \tag{7}$$

Since the evidence selection and relation prediction are scored independently for the FEVER task, we select the parameters using early stopping such that one set of parameters performs the best in terms of evidence recall on the validation set and another performs the best for accuracy.

Although the models are trained jointly to select evidence sentences that help predict the relation to the claim, we may obtain additional improvement by tuning the parameters given the output of the sentence extraction step. In order to do so, we first select the top 5 sentences from the sentence extractor and predict the relation using only those sentences rather than the entire memory as before. In this scenario, we pre-train the model using multi-task learning and tune the parameters for relation prediction while keeping the sentence extraction parameters fixed (and using separate representations for ESIM). We also experimented with a reinforcement learning approach to tune the sentence extractor as in (Chen and Bansal, 2018) but found no additional improvement.

3.5 Experiments

We use Pytorch for our experiments. For the multi-task learning and tuning, we use the Adagrad optimizer with learning rates of 0.01 and 0.001 and gradients clipped to 5 and 2, respectively. For both experiments, we used a batch size of 16. We used the paper development set for early stopping. We initialized the word embeddings with 300-dimensional GloVe vectors and fixed them during training, using a 200-dimensional projection. Out-of-vocabulary words were initialized randomly during both training and evaluation. The second dimension of all other parameter matrices was 200. For the pointer network, we used a beam size of 5 and $q = 3$ hops. λ was set to 1.

	Dev			Test		
	LA	ER	F	LA	ER	F
Base	52.1	44.2	32.6	50.9	45.9	31.9
Gold	68.5	96.0	66.2	65.4	95.2	62.8
MLP	60.4	76.6	51.1	58.7	74.0	49.1
Sep.	56.8	74.9	45.3	53.8	72.7	42.3
MTL	64.0	79.6	55.3	60.5	77.7	52.1
Tune	64.5	79.6	55.8	61.9	77.7	53.2

Figure 4: Paper Development and Test Results (**LA**: Label Accuracy, **ER**: Evidence Recall, **F**: FEVER Score)

4 Results

In Figure 4, we present the sentence extraction, relation prediction, and overall FEVER score for the paper development and test sets. We compare to the baseline (**Base**) from the work of Thorne et al. (2018). For comparison, we also provide the results when using gold document retrieval with a perfect oracle to illustrate the upper bound for our model (**Gold**). First, we illustrate the difference in performance when we train a feedforward network to score the sentences individually (**MLP**) instead of recurrently with a pointer network. In this setting, the sentence extraction in Figure 3 is replaced by a 2-layer feedforward network that is individually applied to every sentence in the memory. The output of the network is a score which is then used to rank the top $l = 5$ sentences. The model is still trained using multi-task learning but with a binary cross-entropy loss in Equation 7 instead of θ_{ptr}. Furthermore, we show the results of the sentence extraction and relation prediction components when trained separately (**Sep.**). We finally present the best results - when trained with multi-task learning (**MTL**) and then tuned (**Tune**).

Our results demonstrate that jointly training a pointer network and relation prediction classifier improves over training separately. We also note that the pointer network, which extracts sentences recurrently by considering the previous sentence, improves over selecting sentences independently using an MLP. Although we obtained improvement by tuning, the improvement is slight, which suggests that the parameter space discovered by multi-task learning is already learning most of the

[2] The evidence as given has no meaningful order but we use the support/refute sequence as provided in the dataset as it may contain annotator bias in terms of importance.

	LA	EF1	F
Paper	62.2	31.6	53.5
Blind	59.7	29.7	49.9
Best	68.2	52.9	64.2

Figure 5: Blind Test Results (**LA**: Label Accuracy, **EF1**: Evidence F1, **F**: FEVER Score)

examples where the model can both identify the correct sentences and label. Finally, we notice that improvements to document retrieval would also improve our model. The gap between **Gold** and **Tune** is around 20 points for evidence recall. When using gold Wikipedia articles (1-2 documents), the number of sentences available (around 20 on average) is less than those selected for our model ($l = 50$), which makes evidence retrieval easier as the memory is smaller. As our models for document retrieval are fairly simple, it is likely that a more complex model could obtain better performance with fewer documents.

Results on the shared task blind test set (prior to any additional human evaluation of the evidence) are presented in Figure 5. For comparison, we show the results on the paper development and test sets (**Paper**) when submitted for the shared task as well as the results of the top system on the leaderboard (**Best**). On the blind test set (**Blind**), overall performance drops by 2-3 points for every metric compared to the paper set. As our analysis in Section 4.1 shows, the performance drops significantly when 2 or more evidence sentences are required. Thus, the performance decrease on the blind test set may be caused by an increase in the number of examples that require additional evidence, although as of this writing the evidence for the test set has not yet been released.

4.1 Analysis

We present an analysis of the performance of the best model (**Tune**) when the model requires multiple pieces of evidence. We present results in Figure 6 for no evidence (for the "not enough info" case) and 1, 2, and 3 or more sentences for label accuracy, evidence recall, and document recall at $k = 5$. When no evidence is required the model only obtains 50% accuracy. We found that this increases to 54% accuracy for the **MTL** model but the accuracy on the other 2 classes decreases. This suggests that using a larger memory improves performance when there is not enough information to make a prediction. Intuitively, reading all relevant Wikipedia articles in their entirety would be necessary for a human to determine this as well. When only 1 sentence is required the model performs well. However, the performance drops significantly on recall when 2 or more sentences are required. This is largely due to the performance of the document retrieval component, which seems to perform poorly when the evidence retrieval requires 2 different Wikipedia articles. When exactly 2 sentences are required, the pointer network retrieves around 60% of the evidence if the retrieved documents are correct. When 3 or more sentences are required, both components perform poorly, suggesting there is significant room for improvement in this case (although there are very few examples requiring this amount of evidence).

Num.	LA	ER	DR
0	50	N/A	N/A
1	74	86	93
2	65	17	26
3+	73	3	14

Figure 6: Paper Development Results by Number of Evidence Sentences Required (**Num.**: Sentences Required, **LA**: Label Accuracy, **ER**: Evidence Recall, **DR**: Document Recall)

5 Conclusion

We presented the results of our system for the FEVER shared task. We described our document retrieval system using lexical and syntactic features. We also described our joint sentence extraction and relation prediction system with multitask learning. The results of our model suggest that the largest gains in performance are likely to come from improvements to detection of the "not enough info" class and retrieval of Wikipedia articles (especially when more than one is required).

For future work, we plan to improve document retrieval and experiment with different sentence representations. Using title features improved document retrieval but for non-Wikipedia data titles would not be available. Furthermore, in other datasets, the titles may not exactly match the text of a claim very often and named entity disambiguation is sometimes needed. One avenue to explore is neural topic modeling trained using article titles (Bhatia et al., 2016).

References

Shraey Bhatia, Jey Han Lau, and Timothy Baldwin. 2016. Automatic labelling of topics with neural embeddings. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 953–963. The COLING 2016 Organizing Committee.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017a. Reading wikipedia to answer open-domain questions. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1870–1879. Association for Computational Linguistics.

Qian Chen, Xiaodan Zhu, Zhen-Hua Ling, Si Wei, Hui Jiang, and Diana Inkpen. 2017b. Enhanced lstm for natural language inference. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1657–1668. Association for Computational Linguistics.

Yen-Chun Chen and Mohit Bansal. 2018. Fast abstractive summarization with reinforce-selected sentence rewriting. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 675–686. Association for Computational Linguistics.

Diederik Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *International Conference on Learning Represen- tations (ICLR)*.

Quoc Le, Ilya Sutskever, Oriol Vinyals, and Lukasz Kaise. 2016. Multi-task sequence to sequence learning. In *International Conference on Learning Represen- tations (ICLR)*.

Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Association for Computational Linguistics (ACL) System Demonstrations*, pages 55–60.

Ramesh Nallapati, Feifei Zhai, and Bowen Zhou. 2017. Summarunner: A recurrent neural network based sequence model for extractive summarization of documents. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA.*, pages 3075–3081.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in pytorch.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Sainbayar Sukhbaatar, Arthur Szlam, Jason Weston, and Rob Fergus. 2015. End-to-end memory networks. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 2440–2448. Curran Associates, Inc.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. Fever: a large-scale dataset for fact extraction and verification. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 809–819. Association for Computational Linguistics.

Oriol Vinyals, Samy Bengio, and Manjunath Kudlur. 2016. Order matters: Sequence to sequence for sets. In *International Conference on Learning Representations (ICLR)*.

Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015. Pointer networks. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 2692–2700. Curran Associates, Inc.